(1)

TASK: UA48
CDRL: 04101
07 July 1992

Training Plan
CARDS

Informal Technical Data

STARS-AC-04101/001/00A
07 July 1992

Paramax Systems
Corporation

12010 Sunrise Valley Drive
Reston VA 22091
Tel 703 620 7000

22 October 1992

PARAMAX
A Unisys Company

STARS-DT-09

Defense Technical Information Center
Bldg. 5 - Cameron Station
Alexandria, VA 22304-6145

Attn: Curtis Gibson

Subject: DTIC Submissions

Enclosures:
1) Ada Formal Methods in the STARS Environment
2) Ada Semantic Interface Specification (ASIS)
3) Integrating Domain-Specific Reuse for System/Software Engineers Course Description
4) PCTE Binding Version Description Document
5) PCTE Browser Tool User Manual
6) PCTE Browser Tool Version Description Document
7) Technical Concept Command Center Library (CARDS)
8) Training Plan (CARDS)

Dear Mr. Gibson:

With regards to our telephone conversation , the above referenced documentation is being submitted to DTIC for public distribution. Each document contains a SF 298 Report Documentation Page, and form 50, DTIC Accession Notice.

There will be a STARS catalog of available products and documentation compiled for the November 16-20 Tri Ada Conference. Paramax would like to have these items included in the STARS catalog. All submission must include a DTIC number. The coordinator for the catalog has established a deadline of October 28 for submissions. Notification of assigned numbers prior to this deadline date would be greatly appreciated.

Please contact the undersigned at (703) 620-7929 if additional information is needed.

Sincerely,

Tanya Lawrence

Tanya Lawrence
STARS Data Management

/tll

TASK: UA48
CDRL: 04101
07 July 1992

INFORMAL TECHNICAL REPORT

For The

SOFTWARE TECHNOLOGY FOR ADAPTABLE, RELIABLE SYSTEMS
(STARS)

Training Plan
Central Archive for Reusable Software
(CARDS)

STARS-AC-04101/001/00
07 July 1992

Data Type: A005, Informal Technical Data

CONTRACT NO. F19628-88-D-0031
Delivery Order 0009

Prepared for:

Electronic Systems Center
Air Force Systems Command, USAF
Hanscom AFB, MA 01731-5000

Prepared by:

EWA
under contract to
Paramax Systems Corporation
12010 Sunrise Valley Drive
Reston, VA 22091

TASK: UA48
CDRL: 04101
07 July 1992

Data ID: STARS-AC-04101/001/00

Distribution Statement "A"
per DoD Directive 5230.24
Authorized for public release; Distribution is unlimited.

Developed by: EWA under contract to
Paramax Systems Corporation

This document, developed under the Software Technology for Adaptable, Reliable Systems (STARS) program, is approved for release under Distribution "A" of the Scientific and Technical Information Program Classification Scheme (DoD Directive 5230.24) unless otherwise indicated. Sponsored by the U.S. Defense Advanced Research Projects Agency (DARPA) under contract F19628-88-D-0031, the STARS program is supported by the military services, SEI, and MITRE, with the U.S. Air Force as the executive contracting agent.

TASK: UA48
CDRL: 04101
07 July 1992

INFORMAL TECHNICAL REPORT
Training Plan
Central Archive for Reusable Software
(CARDS)

Approvals:

Task Manager *Lorraine Martin* *Date*

(Signatures on File)

PREFACE

This Training Plan is designed to function as a guide for conducting domain-specific software reuse training. The Plan addresses training four different audiences. Guidelines are provided on conducting training and developing training materials for Department of Defense (DOD) Organizations, DoD Contractors, System/Software Engineers, and University Professors. The four courses may be implemented either individually or as an integrated package. Feedback and suggestions are welcomed.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	7 July 1992	Informal Technical Report

4. TITLE AND SUBTITLE
Training Plan (CARDS)

5. FUNDING NUMBERS
F19628-88-D-0031

6. AUTHOR(S)
EWA Corporation

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Paramax Corporation
12010 Sunrise Valley Drive
Reston, VA 22091

8. PERFORMING ORGANIZATION REPORT NUMBER
STARS-AC-04101/001/00A

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Department of the Air Force
Headquarters Electronic Systems Division
Hanscom AFB, MA 01731-5000

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
04101

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Distribution "A"

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This Training Plan serves as a comprehensive guide for creating training courses and training materials on domain-specific reuse for Department of Defense (DoD) Organizations, DoD Contractors, System Engineers, and University Professors.

14. SUBJECT TERMS
(CARDS) Central Archive for Reusable Defense Software

15. NUMBER OF PAGES
57

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

TABLE OF CONTENTS

LIST OF FIGURES

1 INTRODUCTION

1.1 PURPOSE

This Training Plan serves as a comprehensive guide for creating training courses and training materials on domain-specific reuse for Department of Defense (DoD) Organizations, DoD Contractors, System Engineers, and University Professors.

1.2 RATIONALE

A foundation for integrating software reuse into the business practices of DoD Organizations and Contractors is necessary if the full potential of software reuse is to be realized. Education and training is the foundation upon which new concepts are incorporated into existing processes. Training and education guidelines must be in place to support the inclusion of reuse in the software development process.

1.3 SCOPE

The Training Plan is a recommendation of how top quality domain-specific software reuse training should be conducted. The Plan identifies the generic functions necessary to support these recommendations. For instance, the Plan itself is detached from dollar figure amounts, but a Price Worksheet (see APPENDIX A) is provided from which these amounts may be derived.

Implementing individual courses will require omitting, adding, and/or modifying the plan's recommendations and outlined course content for different audiences. These changes will ultimately produce a tailored implementation of the plan each time a course is presented to an audience.

The Training Plan provides guidance for conducting four different domain-specific software reuse training courses for four different audiences. Two of the courses are directed at DoD Organizations and DoD Contractors. These two courses are intended to provide an understanding of the management infrastructure required to support software reuse. The third course has been developed to provide system/software engineers with an understanding of how to incorporate software reuse into their current system/software development processes. The fourth course provides outlines on how to incorporate software reuse into university curricula. The intent of the courses in the Training Plan is to demonstrate how software reuse can reduce development and maintenance time and costs, reduce project risks, and increase productivity.

1.4 DEVELOPMENT OF THE TRAINING PLAN

1.4.1 Approach

The purpose of the Training Plan is to specify the approaches to be used in training DoD contractors, university students and professors, and DoD organization personnel in domain-specific reuse. The Training Plan contains outlines (identifying content, methods, and duration) of curriculum contents for each of the target audiences. This approach allows for a wide audience characterization, and contains a tremendous amount of information.

1.4.2 Identification of Courses

In developing the four course outlines, careful consideration was given to the composition of the intended audiences, and how domain-specific reuse impacts their respective functions. The four audiences identified are: management in DoD Organizations, management in DoD Contractors, system/software engineers, and university professors and administrators.

The common business processes that establish a working relationship between DoD Organizations and DoD Contractors were identified to assist in depicting the specific content areas for each of the DoD courses. These processes include developing acquisition plans, requests for proposals (RFPs), statements of work (SOWs), and proposals.

Introducing new technology, such as reuse, may require both managerial and implementation changes throughout all of these business processes. The DoD Organization is identified as initiating the acquisition plans and the RFPs in this working relationship. The DoD Contractor is identified as responding to the DoD Organization's RFP with an appropriate proposal.

The courses for DoD Organizations and DoD Contractors are structured to provide management level personnel with an understanding of reuse and to characterize potential solutions for integrating domain-specific reuse into their particular business practices. The key difference in the courses for these organizations is the focus on their respective views of the business processes and what their individual responsibilities are in making reuse work.

To further refine the content of each course, a distinction was made between managerial related issues and their actual implementation at the technical level. The System/Software Engineer's course was organized to address the technical implementation of reuse. This course is independent of DoD Organization and DoD Contractor distinguishing characteristics at the technical level. Consequently, the course is designed to be applicable to both DoD Organizations and DOD Contractors.

The System/Software Engineer's course objective is to provide a basis for integrating domain-specific reuse into current system/software engineering practices, and to explore how the system/software engineer can use domain analysis products.

The fourth course addressed by this plan is targeted toward university professors. The main objective of this course is to provide an understanding of software reuse, and to identify potential avenues for integrating domain-specific reuse into their courses and curriculum.

1.4.3 Bibliography Used To Develop the Training Plan

Bacon, T.R., and Frierman, L.H. *Shipley Associates Style Guide*, 1990.

Bugelski, B.R. *The Psychology of Learning Applied to Teaching*. Bobbs-Merrill Company, 1964.

Computer Maintenance Training Manual. NSA, M-5099, 1 Oct 1967.

Computer Program Training Plan. Navy-AS, UDI-H-21262, AIR-533, 7 Sep 1973

Dwyer, F.M. *Strategies for Improving Visual Learning.* Learning Services, 1978.

How to Prepare and Conduct Military Training. FM 21-6. HQ, Department of the Army, Nov 1975.

Proposed Curricula for Librarians Course. Asset Source for Software Engineering Technology (ASSET), National Software Technology Repository.

STARS Reuse Concept of Operation. Task US30: Vol I Ver 0.5 - DRAFT STARS-SC-03725/001/00A, 27 Aug 1991.

Statement of Work for Electronic Warfare Associates. Task UA48: Central Archive for Reusable Defense Software (CARDS), Phase II, 13 Feb 1992.

Steinger, L. *The Career Development Program for Acquisition Personnel.* DoD Report 5000.52-M, 15 Nov 1991.

Student and Training Course Evaluation Forms. CDRL Item A028, Contract F19630-88-D-0004.

Student's Training Course Guide. DoD, DI-H-7071, NAVAIE 04B1, 8 Feb 1981.

Student Training Materials. NTEC, DI-H-25724B, NAVTRAEQUIPCEN Code N-25, 30 Jun 1986.

Tomayko, J.E. *Software Configuration Management.* SEI Curriculum Module SEI-CM-4-1.3 (Preliminary), Jul 1987.

Training and Equipment Plan. DoD, DI-H-7066, NAVAIR 041B, 18 Feb 1981.

Training Development and Support Plan. F/AFSPACECOM-1013 CCTS, DI-MGMT-80476, 26 Aug 1987.

Warriner's English Grammar and Composition, Fifth Course. New York: Harcourt Brace Jovanovich, 1982.

1.4.4 Individual Course Information

For each of the four training courses, the plan includes a section on course characteristics, an outline of the course content, recommended instructor qualifications, and references used to develop the course.

The section on course characteristics provides a course overview (including a rationale for conducting the training), the course objectives, the course's completion criteria, an audience characterization, and a student reading list.

The instructor should use the student reading list to develop suggested prerequisite readings, course texts, and supplemental readings. The supplemental reading list should

provide students with a collection of documents that will be useful in advancing their understanding of domain-specific reuse beyond the information covered in the course.

Training Course
for DoD Organizations

2 TRAINING COURSE FOR DoD ORGANIZATIONS

The DoD Organization is identified as initiating acquisition plans and RFPs with DoD Contractors. The purpose of this course is to provide DoD Organization management-level personnel with an understanding of the infrastructure required to support the integration of domain-specific reuse.

2.1 COURSE CHARACTERISTICS

2.1.1 Rationale

The DoD supports a focus toward more cost-effective and timely development of software-intensive systems. Reuse of software system components is being promoted as supporting this objective. Domain-specific reuse provides procedures and methods to effectively reuse common domain components. This course will provide reference for integrating domain-specific reuse into the management processes of a DoD Organization.

2.1.2 Course Objectives

The learning objective for this course is to provide DoD Organization management level personnel with:

an understanding of the history of reuse and domain-specific reuse;

an understanding of the responsibilities involved in providing a reuse infrastructure;

an understanding of how reuse may impact writing acquisition plans, RFPs and SOWs, and evaluating submitted proposals;

an understanding of how reuse can be integrated into current processes; and

potential ways to integrate domain-specific reuse into a DoD Organization.

2.1.3 Course Completion Criteria

Participants, working in small interactive groups, will apply concepts and techniques identified by the course's content areas, producing products as proof of completing the training course.

Upon completion of the training course the participants will be able to:

identify areas within their DoD organizations that should support reuse;

outline specific procedures for implementing reuse;

develop plans for identifying applicable domain areas and reuse infrastructures for their mission;

incorporate reuse concepts into RFPs, and SOWs, and

develop reuse-oriented evaluation criteria for proposals.

2.1.4 Audience Characterization

The intended audience for this training course is management personnel in a DoD organization, especially direction-level managers and program managers.

Direction-level managers are those responsible for implementing cross-program infrastructure change, instantiating top-level policy guidance and change, managing program costs, and instituting policy change in procurement practices and/or system development methods.

Direction-level managers may also possess an interest in domain-specific reuse, management level awareness of technology, an interest in cost and benefit issues, an interest in long term benefits, and an awareness of their application domain.

Program managers are responsible for planning and executing acquisition programs, instantiating program level policy guidance/change, and risk assessment, analysis, and mitigation.

Program managers may possess an interest in domain-specific reuse, knowledge of basic system acquisition, knowledge about general software development issues, management level awareness of technology, and an awareness of cost, schedule, and requirements.

This course is intended for people whose primary responsibility is strategic management of their business.

2.1.5 Student Readings

Arnold, R.S., Frakes, W., and Prieto-Diaz, R. *Software Reuse, Domain Analysis, and Reengineering*. Conference Notes, 6-8 Apr 1992.

STARS Reuse Concept of Operations. STARS-SC-03725/001/00A, Vol I Ver 5, 27 Aug 1991.

US45 task for STARS Contractual Environment Guidance. US45 Task for STAR, Paramax.

US45 task for STARS Reusable Software Acquisition Environment Guidebook (DRAFT). US45 Task for STARS, Paramax.

Williams, A. *SWAN - An Ada Program for Cost Estimation*. AdaIC Newsletter, Sep 1991.

Wong, W. *A Management Overview of Software Reuse*. NBS Special Publication 500-142, Washington, DC: USGPO, Sep 1983.

2.2 OUTLINE OF RECOMMENDED COURSE CONTENT

The following is an outline of the recommend course content for the DoD Organizations Course.

2.2.1 Reuse as Part of the Solution

A. The Dilemma

1. Software industry

   a. Increasing software costs versus hardware costs
   b. Increasing complexity of software
   c. Increasing development time
   d. Shortage of experienced personnel

2. Customer

   a. Demand for high quality products
   b. Demand for low risk products
   c. Demand for automated programming support
   d. Demand for customized software packages

B. The Solution

1. What is Reuse?

   a. The process of incorporating into the life-cycle of a software system any preexisting components (e.g, requirements, design, code, executables).
   b. Examples

2. Why Reuse?

   a. Reduces schedule
   b. Increases productivity

      (1) Amplifies programming capabilities
      (2) Reduces the amount of documentation and testing

   c. Increases quality

      (1) Software is well designed (for reuse)
      (2) Software is well documented (standard)
      (3) Software is well tested (certified)
      (4) Software is well understood (functionality)
      (5) Concepts are manifested in rapid prototyping

   d. Increases reliability
   e. Reduces maintenance costs
   f. Overall, reuse allows for more system development within the same time frame
   g. Reduces long term development costs

3. Evolution of Reuse

   a. Early reuse projects
   b. Reuse success stories

c. Current approaches and concepts

(1) Components and wide spectrum reuse
(2) Composition versus generation
(3) Design for reuse
(4) Design with reuse
(5) Life-cycle importance of reuse

2.2.2 Introduction to Domain-Specific Reuse

A. What is a Domain?

1. A domain is a set of common capabilities and data which constitute a set of current and future systems in a particular application area, such as:

a. A business area
b. A software business area
c. A software intensive application area
d. An application area for which similar software systems have been built

2. Examples

B. What is Domain-specific Reuse?

1. Reuse of ideas, knowledge, artifacts, personnel, and components in an existing domain.
2. Examples

C. Why Domain-specific reuse?

1. Reuse is more effective in a narrow, well defined domain where similar systems are built.
2. Examples

D. Domain Selection Checklist

1. Define strategy.
2. Select domain analysis resources.
3. Identify boundaries/scope of domain.
4. Select mature, stable, well-defined domain.
5. Define predictable technology.
6. Perform market evaluation.
7. Identify available domain expertise.
8. Conduct readiness cost and benefit analysis.

E. Existing Technological Support

1. STARS efforts
2. CARDS
3. PRISM
4. RAPID
5. Others

2.2.3 Considerations When Integrating Reuse

A. Strategic Planning

1. Cost-benefit Analysis

An organization must consider the costs and benefits of implementing reuse. Some of the questions that can be asked are:

a. How much does reuse cost?
b. Is a reuse program economically feasible?
c. What are the alternatives to a reuse program?
d. What alternatives exist for implementing a reuse program (technical support)?
e. What is the scope of the reuse program?
f. At what organizational level is the program targeted?
g. Some of the existing software cost models are :

(1) SPC
(2) Reuse COCOMO
(3) Ada/COCOMO
(4) SoftCost
(5) IDA/STARS
(6) REVIC

2. Feasibility Analysis

An organization must decide if it is feasible to incorporate reuse into their current business practices. Some of the questions that can be asked to decide this strategic factor are:

a. How many similar systems will be built?
b. Is reuse beneficial?
c. Does the organization want to incoporate reuse?
d. Does the organization have the resources?
e. Is management committed?
f. Are implementation variations small or large?
g. Is existing software already available for reuse?
h. Is software production large enough to justify a reuse program?

3. Domain Suitability

An organization must determine if its current operational domain is suitable for reuse. Following are some of the questions to help answer this issue:

a. Is the domain broad or narrow?
b. Is the domain mature and well understood?
c. Is the domain stable or changing continually?
d. Is the domain based on well established principles, methods, and formalisms?

4. Legal/Acquisition Issues

   a. Ownership/Copyright/Proprietary issues
   b. Liabilities and responsibilities

      (1) Domain Model
      (2) Domain Architecture
      (3) Components

   c. Contractual requirements
   d. Component content

B. Implementation Planning

1. Long-term Commitment to Reuse
2. Strong Management Support
3. Instantiate a Reuse Infrastructure

   a. Stable: the same structure supports all stages of the same program.
   b. Flexible: the roles and people can be changed without affecting function.
   c. Evolving: reuse may start with a minimum set of one person in multiple roles and evolve into multiple teams with specific roles.
   d. Practical: a reuser can actually practice reuse.
   e. Effective: to the extent that all the elements of the infrastructure are efficient.
   f. Economical: cost, complexity, and sophistication are adjustable to available budget.

4. Develop Investment Strategy

   a. Initial financial investment

      (1) Resources

         (a) Hire experienced reuse personnel
         (b) HW/SW purchases
         (c) Obtaining and testing components
         (d) Developing components

      (2) Education/training programs

         (a) Management level
         (b) Technical

      (3) Setting up a reuse infrastructure

         (a) Developing a domain model
         (b) Developing an architectural model
         (c) Developing a library model
         (d) Building and maintaining a library

(4) Supporting the reuse infrastructure (library/technical) support

5. Allow Time for Transition
6. Provide Continuous Financial Support

a. Resources
b. Education/training
c. Library/technical
d. Monitor reuse community
e. Long-term investment

7. Initiate Incentive Programs

a. At all levels
b. Especially engineers

8. Develop Accommodating Business Practices

a. Reusable Software Acquisition Factors

(1) Identify effects on current acquisition policies
(2) Incorporate reuse into existing acquisition policies
(3) Introduce reusable software acquisition guidebooks and handbooks

b. Developing RFPs which Incorporate Reuse

(1) Locating and evaluating components including domain models and architectural models
(2) Listing applicable government reuse libraries
(3) Listing COTS products
(4) Encourage continuous identification of additional library products
(5) Identify status of software rights
(6) Criteria for award based on reuse

c. Evaluating Proposals Involving Reuse

(1) Requirements
(2) Specifications
(3) Additional expected costs
(4) Added benefits
(5) Risk reduction
(6) Indication of technical expertise in approach
(7) Proposal evaluation criteria

d. Developing SOWs which Mandate Reuse

(1) Management related tasks regarding subcontracting software
(2) Impact life-cycle development
(3) Document contract/legal issues

(4) Identify reuse libraries
(5) Define personnel requirements

9. Develop a Reuse Measurement Plan

a. Reuse level: the percent of a system made up of reused components
b. Quality: the fitness of software for reuse
c. Productivity: how much software is produced per unit effort? (e.g., lines/person-month)
d. Metrics: for evaluating components and systems

10. Address Resistance During Transition

a. Social/cultural biases
b. Confusion with new technology
c. Loss of experienced personnel
d. Team effort required

11. Incorporating Reuse into Your Organization

a. Incrementally staged
b. Formal
c. Systematic

C. Technical Issues

1. Domain-analysis processes
2. Effects on current processes
3. Technology support

2.2.4 Continuing Education

A. Monitor Reuse Community Activities

B. Financial Commitment

2.2.5 Lessons Learned

A. Points Confirmed

1. Reuse program changes the software development process
2. Technology is important, but not essential
3. Reuse is more effective in narrow, well defined domains
4. Infrastructure support is essential
5. Classification is instrumental in domain understanding
6. Reuse is financially successful

B. Factors Which Contribute to Failure

1. Lack of management support

2. No incentives
3. No procedures
4. Not enough information in library component catalog
5. Poor classification
6. No automated library
7. Original parts not designed for reuse

2.2.6 Demonstration of Reuse Library

A. Connecting to the Library

B. Log-In Procedures

C. Component Selection and Retrieval Mechanisms

D. Log-Out Procedures

2.3 TRAINING INSTRUCTOR

This section outlines the responsibilities and desired qualifications of the Training Instructor for the DoD Organizations Course. These optimum qualifications are only recommendations; they describe the best candidate for this position. In the event that it is not possible to locate an individual who meets all of the detailed qualifications, some of the qualifications may be relaxed.

2.3.1 Job Description

The Training Instructor is responsible for conducting a tailored implementation of the recommended course content through a lecture/workshop format.

This is accomplished by completing the following tasks:

consulting the documents listed in Section 2.4, Bibliography for Developing DoD Organization Course Content;

identifying specific implementor-related actions for reuse integration; updating and tailoring the recommended outline of course content with respect to DoD Organizations;

preparing appropriate training materials;

estimating the duration of the training session based on a blend of optimum cost-effectiveness and covering the course content thoroughly;

conducting the training session;

seeing that the atmosphere of the training session is informal and relaxing, intellectually stimulating, and learner-centered;

integrating lecture, discussion, and small working groups into the training session;

leading the course completion criteria workshop; and

providing an evaluation of the course completion criteria.

2.3.2 Formal Education

The Instructor should hold at least a BS degree in business administration (an MS degree is preferred) and a degree in computer science, system engineering, or a closely related field.

2.3.3 Knowledge of Instruction

The Instructor should possess an understanding of the principles of learning, the methods of teaching, an ability to apply these principles and methods, and excellent communication skills.

2.3.4 Practical Teaching Experience

The Instructor should possess teaching experience (at least two years), in planning short courses and seminars, and in addressing DoD management level personnel.

2.3.5 Knowledge of Subject

The Instructor should possess:

experience in the DoD's contractual and management processes;

knowledge and an understanding of the impact of integrating reuse;

experience in DoD procurement practices, including specific experience in writing RFPs and SOWs, and evaluating proposals;

work experience at the DoD organization management level;

practical experience in reuse library operations; and

work experience in software development.

2.4 BIBLIOGRAPHY FOR DEVELOPING DoD ORGANIZATION COURSE CONTENT

The Training Instructor should consult the following documents to prepare the tailored implementation of the course content and the training materials.

Arnold, R.S., Frakes, W., and Prieto-Diaz, R. *Software Reuse, Domain Analysis, and Reengineering*. Conference Notes, 6-8 Apr 1992.

Matsumoto, Y. *Some Experience in Promoting Reusable Software: Presentation in Higher Abstract Levels.* IEEE Transaction on Software Engineering, Vol SE-10 No 5, Sep 1984.

Tracz, W. *Ada Reusability Efforts: A Survey of the State of the Practice.* Proceedings of the Joint Ada Conference, Fifth National Conference on Ada Technology and Washington Ada Symposium, US Army Communications-Electronics Command, Fort Monmouth, New Jersey.

US45 task for STARS Contractual Environment Guidance. US45 Task for STARS, DSD Laboratories, Inc.

US45 task for STARS Reusable Software Acquisition Environment Guidebook (DRAFT). US45 Task for STARS, DSD Laboratories, Inc.

Williams, A. *SWAN - An Ada Program for Cost Estimation.* AdaIC Newsletter, Sep 1991.

Training Course
for DoD Contractors

3 TRAINING COURSE FOR DoD CONTRACTORS

The DoD Contractor is identified as responding to a DoD Organization's RFP with an appropriate proposal. The purpose of this course is to provide DoD Contractor management-level personnel with an understanding of the infrastructure required to support the integration of domain-specific reuse.

3.1 COURSE CHARACTERISTICS

3.1.1 Rationale

The DoD supports a focus toward more cost-effective and timely development of software-intensive systems. Reuse of software system components is being promoted as supporting this objective. Domain-specific reuse provides procedures and methods to effectively reuse common domain components. This course will provide reference for integrating domain-specific reuse techniques into the management processes of a DoD Contractor.

3.1.2 Course Objectives

The learning objectives for this course are to provide DoD Contractor management level personnel with:

an understanding of domain-specific reuse;

an understanding of the responsibilities involved in providing a reuse infrastructure;

the ability to respond to RFPs with proposals that incorporate reuse; and

an understanding of the potential methods for integrating domain-specific reuse into current business practices.

3.1.3 Course Completion Criteria

Participants, working in small interactive groups, will apply concepts and techniques identified by the course's content areas, producing products as proof of completing the training course.

Upon completion of the training course the participants will be able to:

draft an organization chart identifying areas within their organizations that should emphasize reuse;

outline specific procedures for implementing reuse;

develop plans for identifying applicable domain areas within their business;

identify strategies for incorporating reuse into proposals,

develop reuse-oriented evaluation criteria in relation to RFPs and SOWs.

3.1.4 Audience Characterization

The intended audience for this training course DoD Contractor management personnel.

DoD Contractor management personnel are those responsible for implementing cross project infrastructure change; initiating top-level policy guidance and change; managing project costs; and instituting policy change in marketing practices and/or system development methods. Managers may also possess an interest in domain-specific reuse, management level awareness of technology, an interest in cost and benefit issues, an interest in long term benefits, and an awareness of their application domain.

This course is intended for people whose primary responsibility is strategic management of their business.

3.1.5 Student Readings

Arnold, R.S., Frakes, W., and Prieto-Diaz, R. *Software Reuse, Domain Analysis, and Reengineering*. Conference Notes, 6-8 Apr 1992.

STARS Reuse Concept of Operations. STARS-SC-03725/001/00A, Vol I Ver 5, 27 Aug 1991.

US45 task for STARS Contractual Environment Guidance. US45 Task for STAR, Paramax.

US45 task for STARS Reusable Software Acquisition Environment Guidebook (DRAFT). US45 Task for STARS, Paramax.

Williams, A. *SWAN - An Ada Program for Cost Estimation*. AdaIC Newsletter, Sep 1991.

Wong, W. *A Management Overview of Software Reuse*. NBS Special Publication 500-142, Washington, DC: USGPO, Sep 1983.

3.2 OUTLINE OF COURSE CONTENT

The following is an outline of the recommended course content for the DoD Contractors Course.

3.2.1 Reuse as Part of the Solution

A. The Dilemma

1. Software industry

a. Increasing software costs versus hardware costs
b. Increasing complexity of software

c. Increasing development time
d. Shortage of experienced personnel

2. Customer

   a. Demand for high quality products
   b. Demand for low risk products
   c. Demand for automated programming support
   d. Demand for customized software packages

B. The Solution

1. What is Reuse?

   a. The process of incorporating into the life-cycle of a software system any preexisting components (e.g, requirements design, code, executables).
   b. Examples

2. Why Reuse?

   a. Reduces schedule
   b. Increases productivity

      (1) Amplifies programming capabilities
      (2) Reduces the amount of documentation and testing

   c. Increases quality

      (1) Software is well designed (for reuse)
      (2) Software is well documented (standard)
      (3) Software is well tested (certified)
      (4) Software is well understood (functionality)
      (5) Concepts are manifested in rapid prototyping

   d. Increases reliability
   e. Reduces maintenance costs
   f. Overall, reuse allows for more system development within the same time frame
   g. Reduces long-term development costs

3. Evolution of Reuse

   a. Early reuse projects
   b. Reuse successes
   c. Current approaches and concepts

      (1) Components and wide spectrum reuse
      (2) Composition versus generation
      (3) Design for reuse
      (4) Design with reuse
      (5) Life-cycle importance of reuse

3.2.2 Introduction to Domain-Specific Reuse

A. What is a Domain?

1. A domain is a set of common capabilities and data which constitute a set of current and future systems in a particular application area, such as:

   a. A business area
   b. A software business area
   c. A software intensive application area
   d. An application area for which similar software systems have been built

2. Examples

B. What is Domain-specific Reuse?

1. Reuse of ideas, knowledge, artifacts, personnel, and components in an existing domain.
2. Examples

C. Why Domain-specific reuse?

1. Reuse is more effective in a narrow, well defined domain where similar systems are built.
2. Examples

D. Domain Selection Checklist

1. Define strategy.
2. Select domain analysis resources.
3. Identify boundaries/scope of domain.
4. Select mature, stable, well-defined domain.
5. Define predictable technology.
6. Perform market evaluation.
7. Identify available domain expertise.
8. Conduct readiness cost and benefit analysis.

E. Existing Technological Support

1. STARS efforts
2. CARDS
3. PRISM
4. RAPID
5. Others

3.2.3 Considerations When Integrating Reuse

A. Strategic Planning

1. Cost-benefit Analysis

   An organization must consider the costs and benefits of implementing reuse. Some of the questions that can be asked are:

   a. How much does reuse cost?
   b. Is a reuse program economically feasible?

c. What are the alternatives to a reuse program?
d. What alternatives exist for implementing a reuse program (technical support)?
e. What is the scope of the reuse program?
f. At what organizational level is the program targeted?
g. Some of the existing software cost models are:

(1) SPC
(2) Reuse COCOMO
(3) Ada/COCOMO
(4) SoftCost
(5) IDA/STARS
(6) REVIC

2. Feasibility Analysis

An organization must decide if it is feasible to incorporate reuse into their current business practices. Some of the questions that can be asked to decide this strategic factor are:

a. How many similar systems will be built?
b. Is reuse beneficial?
c. Does the organization want to incorporate reuse?
d. Does the organization have the resources?
e. Is management committed?
f. Are implementation variations small or large?
g. Is existing software already available for reuse?
h. Is software production large enough to justify a reuse program?
i. Will the organization be more attractive to DoD if reuse is incorporated into current business practices?
j. Will the organization be more competitive if reuse is incorporated into current business practices?

3. Domain Suitability

An organization must determine if its current operational domain is suitable for reuse. Following are some of the questions to help answer this issue:

a. Is the domain broad or narrow?
b. Is the domain mature and well understood?
c. Is the domain stable or changing continually?
d. Is the domain based on well established principles, methods, and formalisms?

4. Legal/Acquisition Issues

a. Ownership/Copyright/Proprietary issues
b. Liabilities and responsibilities
c. Contractual requirements
d. Component content

B. Implementation Planning

1. Long-term Commitment to Reuse
2. Strong Management Support
3. Instantiate a Reuse Infrastructure

    a. Stable: the same structure supports all stages of the same program
    b. Flexible: the roles and people can be changed without affecting function
    c. Evolving: reuse may start with a minimum set of one person in multiple roles and evolve into multiple teams with specific roles
    d. Practical: a reuser can actually practice reuse
    e. Effective: to the extent that all the elements of the infrastructure are efficient
    f. Economical: cost, complexity, and sophistication are adjustable to available budget

4. Develop Investment Strategy

    a. Initial financial investment

        (1) Resources

            (a) Hire experienced reuse personnel
            (b) HW/SW purchases
            (c) Obtaining and testing components
            (d) Developing components

        (2) Education/training programs

            (a) Management level
            (b) Technical

        (3) Setting up a reuse infrastructure
        (4) Supporting the reuse infrastructure (library/technical) support

5. Allow Time for Transition
6. Provide Continuous Financial Support

    a. Resources
    b. Education/training
    c. Library/technical
    d. Monitor reuse community
    e. Long-term investment

7. Initiate Incentive Programs

    a. At all levels
    b. Especially engineers

8. Develop Accommodating Business Practices

    a. Evaluating RFPs Which Specify Reuse

(1) Identifying applicable government reuse libraries
(2) Locating COTS products
(3) Identifying additional library products
(4) Identify status of software rights
(5) Understanding criteria for award based on reuse

b. Developing Proposals Incorporating Reuse

(1) Requirements
(2) Specifications
(3) Additional expected costs
(4) Added benefits
(5) Risk reduction
(6) Indication of technical expertise in approach
(7) Proposal evaluation criteria

c. Evaluating SOWs Which Require Reuse

(1) Understand management related tasks regarding subcontracting software
(2) Impact on life-cycle development
(3) Evaluate contract/legal issues
(4) Identify reuse libraries
(5) Evaluate personnel requirements

9. Develop a Reuse Measurement Plan

a. Reuse level: the percent of a system made up of reused components
b. Quality: the fitness of software for reuse
c. Productivity: how much software is produced per unit effort? (e.g., lines/person-month)
d. Metrics: for evaluating components and systems

10. Address Resistance During Transition

a. Social/cultural biases
b. Confusion with new technology
c. Loss of experienced personnel
d. Team effort required

11. Adoption of Reuse

a. Incrementally staged
b. Formal
c. Systematic

C. Technical Issues

1. Domain-Analysis processes
2. Effects on current processes
3. Technology support

3.2.4 Continuing Education

A. Monitor Reuse Community Activities

B. Financial Commitment

3.2.5 Lessons Learned

A. Points Confirmed

1. Reuse program changes the software development process.
2. Technology is important, but not essential.
3. Reuse is more effective in narrow, well defined domains.
4. Infrastructure support is essential.
5. Classification is instrumental in domain understanding.
6. Reuse is financially successful.

B. Factors which Contribute to Failure

1. Lack of management support
2. No incentives
3. No procedures
4. Not enough information in catalog
5. Poor classification
6. No automated library
7. Original parts not designed for reuse

3.2.6 Demonstration of Reuse Library

A. Connecting to the Library

B. Log-In Procedures

C. Component Selection and Retrieval Mechanisms

D. Log-Out Procedures

3.3 TRAINING INSTRUCTOR

This section outlines the responsibilities and desired qualifications of the Training Instructor for the DoD Contractors Course. These optimum qualifications are recommendations; they describe the best candidate for this position. Should it not be possible to locate an individual who meets all of the detailed qualifications, some of the qualifications may be waived.

3.3.1 Job Description

The Training Instructor is responsible for conducting a tailored implementation of the recommended course content through a lecture/workshop format.

This is accomplished by completing the following tasks:

consulting the documents listed in Section 3.4, Bibliography for Developing DoD Contractor Course Content;

identifying specific implementor-related actions for reuse integration; updating and tailoring the recommended outline of course content with respect to the DoD Contractor;

preparing appropriate training materials;

estimating the duration of the training session based on a blend of optimum cost-effectiveness and covering the course content thoroughly;

conducting the training session;

seeing that the atmosphere of the training session is informal and relaxing, intellectually stimulating, and learner-centered;

integrating lecture, discussion, and small working groups into the training session;

leading the course completion criteria workshop; and

providing an evaluation of the course completion criteria.

3.3.2 Formal Education

The Instructor should hold at least a BS degree in business administration (an MS degree is preferred) and a degree in computer science, system engineering, or a closely related field.

3.3.3 Knowledge of Instruction

The Instructor should possess an understanding of the principles of learning, the methods of teaching, an ability to apply these principles and methods, and excellent communication skills.

3.3.4 Practical Teaching Experience

The Instructor should possess teaching experience (at least two years), in planning short courses and seminars, and in addressing management level personnel.

3.3.5 Knowledge of Subject

The Instructor should possess:

experience in the DoD's contractual and management processes;

knowledge and an understanding of the impact of integrating reuse;

experience in DoD procurement practices, including specific experience in evaluating RFPs, and SOWs, and writing proposals;

work experience at the DoD contractor management level;

practical experience in reuse library operations; and

work experience in software development.

3.4 BIBLIOGRAPHY FOR DEVELOPING DoD CONTRACTOR COURSE CONTENT

The Training Instructor should consult the following documents in preparing the tailored implementation of the course content and the training materials.

Arnold, R.S., Frakes, W., and Prieto-Diaz, R. *Software Reuse, Domain Analysis, and Reengineering*. Conference Notes, 6-8 Apr 1992.

Matsumoto, Y. *Some Experience in Promoting Reusable Software: Presentation in Higher Abstract Levels*. IEEE Transaction on Software Engineering, Vol SE-10 No 5, Sep 1984.

Tracz, W. *Ada Reusability Efforts: A Survey of the State of the Practice*. Proceedings of the Joint Ada Conference, Fifth National Conference on Ada Technology and Washington Ada Symposium, US Army Communications-Electronics Command, Fort Monmouth, New Jersey.

US45 task for STARS Contractual Environment Guidance. US45 Task for STARS, DSD Laboratories, Inc.

US45 task for STARS Reusable Software Acquisition Environment Guidebook (DRAFT). US45 Task for STARS, DSD Laboratories, Inc.

Williams, A. *SWAN - An Ada Program for Cost Estimation*. AdaIC Newsletter, Sep 1991.

Training Course
for System and Software Engineers

4 TRAINING COURSE FOR SYSTEM AND SOFTWARE ENGINEERS

The purpose of this course is to provide system and software engineers with an introduction to system development with domain-specific software reuse. The course introduces the methods necessary to integrate domain-specific software reuse concepts into current system and software development processes by emphasizing domain analysis, generic architecture development, instantiation of the generic architecture, and system composition.

4.1 COURSE CHARACTERISTICS

4.1.1 Rationale

This course provides the support for the integration of domain-specific reuse into system and software development processes. To facilitate domain-specific software reuse, it is necessary to provide system and software engineers with an understanding of methods and techniques necessary to perform domain-specific software reuse activities.

4.1.2 Course Objective

The course objectives are: to provide a basis for integrating domain-specific reuse into current system and software engineering practices, and to explain how the system/software engineer can use domain analysis products for the modeling, simulation, and prototyping of a system.

Although it is necessary to cover domain analysis techniques, the emphasis of this course is in the use of domain analysis products to support domain-specific reuse in application development.

4.1.3 Course Completion Criteria

The following products will be produced as proof of course participation:

A context model for a simple domain.

A domain model for a simple domain.

A specific system architecture from a generic architecture.

A system prototype using a domain-specific software reuse library.

4.1.4 Audience Characterization

This course is intended for skilled system and software engineers. It is not intended for introductory level computer programmers.

System engineers, called system analysts in some application domains, are concerned with the decomposition of systems, the allocation of software development responsibility

for specific system components (to specialist developers or manufacturers), and with the subsequent composition of software and hardware system components to produce the final system.

System engineers begin with customer-defined goals and constraints and derive a representation of function, performance, interfaces, design constraints, and information structure that can be allocated to each of the generic system elements.

Software engineers are concerned with how the individual software components are actually designed, developed, and integrated into the total system.

4.1.5 Student Readings

Arango, G. and Prieto-Diaz, R. *Domain Analysis and Software Systems Modeling*. IEEE Computer Society Press, May 1991.

Biggerstaff, T. *Topics In Reuse and Design.* IEEE Video, 1991.

Cohen, S. *Software Reuse Technology: Feature-Oriented Domain Analysis.* SEI Tutorial slides, Mar 1992.

Freeman, P. *Software Reusability.* IEEE Computer Society Press, 1987.

Gause, D.C., and Weinberg, G.M. *Exploring Requirements Quality Before Design.* Dorset House Publishing, 1989.

Glass, R.L. *Building Quality Software.* Prentice Hall, 1992.

Pressman, R.S. *Software Engineering: A Practitioner's Approach.* McGraw-Hill, 1992.

Tracz, W. *Software Reuse - Emerging Technology.* IEEE Computer Society Press, Sep 1985.

Withey, J. *Model-Based Engineering.* SEI Tutorial slides, Mar 1992.

Yourdon, E. *Modern Structured Analysis.* Yourdon Press Computing Series, Prentice-Hall, 1989.

4.2 OUTLINE OF COURSE CONTENT

4.2.1 Introduction to Reuse

It is necessary to convince the system and software engineers that by making domain-specific reuse a part of their system specification process, they will lower risk and build better systems.

A. Incentives for Reuse

1. Customer demand issues
   a. Customized software packages
   b. Low risk products

c. High quality products
d. Fast delivery

2. Reduce system development time

By using reusable components, the complexity of the system can be addressed and development time decreased.

3. System Reliability is Increased

a. Components which have been previously proven in working system are more reliable than untested components.
b. System testing time is reduced.

4. Overall Risk is Reduced

a. Greater uncertainty exists in the costs associated with developing a new component.
b. There is an initial investment associated with reusing an existing component.

B. Reuse and the Software Life-cycle

1. Design for Reuse

Build into the design and implementation the flexibility, variations and adaptability to create a reusable component.

2. Design with Reuse

Incorporate available reusable assets into system analysis and design.

3. Entry-points of Life-cycle Artifacts

It is important to stress that reusable components include all life-cycle artifacts such as requirements, design, code, and test suites.

C. Evolution of Domain-specific Software Reuse

1. Domain-specific Software Reuse

A higher return on reuse is realized when requirements are reused rather than just code. The instructor will introduce the student to the concept of domain product reuse. The instructor needs to explain the basic concepts and advantages of domain-specific reuse. This is just a top-level introduction; the detailed presentation follows.

2. Pioneering Software Reuse Projects

a. CAMP
b. Raytheon Missile Systems Division

c. GTE DS Asset Management Program

D. Payoffs of Domain-specific Software Reuse

Comparison of building a system architecture from scratch as opposed to building it using domain-specific assets

1. Decrease system development time
2. Increase productivity
3. Increase quality
4. Increase maintainability
5. Increase job performance

E. Software Reuse Libraries

1. Library as a Reuse Tool

The instructor will introduce the reuse library as a tool, not as the reuse solution.

2. Reuse Library Representations

The instructor will introduce a brief overview of various methods (e.g., Dewey decimal, semantics, faceted, indexed, and keywords).

3. Search and Retrieval Methods

The instructor will have the students consider full text searching, facet searching, keyword searching, knowledge-based searching, and browsing.

4. Available Libraries

The instructor will provide the students with an overview of several different types of libraries and the purpose each serves the reuse community.

a. CARDS
b. ASSET
c. RAPID
d. DSRS
e. AdaNET
f. COSMIC
g. ASR
h. Others

4.2.2 Domain Analysis

The teaching of domain analysis will be complete enough to instill an awareness of domain analysis techniques and their role in systematic software reuse. This section will also build a trust in the products produced by the domain analysis process.

A. Domain Modeling Methodolgies

Most domain modeling methodologies formulate similar models and viewpoints of the domain. These models can often include feature models, functional models, object models, and composition rules. Some of the common modeling techniques used are Entity-Relationship model, State diagrams, hierarchical model and other structured analysis design techniques. The instructor will review several common structured analysis and modeling techniques. A brief overview of several domain modeling methodologies will be presented.

B. Domain Identification

It is necessary to choose the right domains to analyze. A domain must be stable, well understood and somewhat static to be a good candidate for domain analysis. A discussion and an illustration of how a domain is chosen will occur.

C. Domain Context Analysis

The boundaries of the domain must be properly scoped. This process places the domain relative to other domains. This process also defines the external entities and data flows between the external entities and the domain.

Lab: The instructor will provide the students with the name and a brief description of a domain. The students will create a context diagram of the domain specified.

D. Domain Modeling

The domain model captures the common parts of the domain along with the differences. The data commonality and the control flow for the functionality is captured in a functional model. This functional model is then used to generate the requirements for a generic architecture and the reusable components.

Lab: The students will create a functional model of the domain specified by the instructor.

E. Domain Analysis Products

Using the models, composition rules, and taxonomy generated by domain analysis, a generic architecture is developed. The generic architecture provides interface specification among the components of the domain model. In developing the generic architecture, software constraints, hardware constraints, performance constraints and general design constraints are all considered. The student must understand the structure of the generic architecture in order to use it.

4.2.3 Integration of Reuse Into System Development Processes

In order to achieve reuse at the system composition level, the system engineer must understand how the various domain products integrate into the system development process at the requirements definition level.

This section will use instructor-provided domain models, system requirements and a domain-specific library to illustrate and support the development of a specific system architecture and a system prototype. This is an intensive hands-on part of the course.

A. Domain-specific Libraries

A domain-specific reuse library is used as a tool to model the domain and build a system. The instructor will review some of the earlier library concepts and any additional considerations necessary to build and use a domain-specific library.

B. Instantiation of the Generic Architecture

Using a generic architecture and reuse library, the student will build a specific system architecture from an instructor-provided generic architecture.

The instructor will use this activity to:

1. Demonstrate to the students how to use the domain-specific reuse library.

2. Illustrate to the students how the generic domain analysis products can clarify requirements.

3. Show how the generic domain analysis products can be used to define requirements.

Lab: Design a specific system architecture by using the simple domain model and generic architecture.

C. Generic Artifacts and System Composition

The software components contained within the software reuse library have been designed in a manner that enables them to be reused without detailed knowledge of the code itself. These components can be assembled to create a prototype. The majority of the course time will be spent doing this exercise.

Lab: The students will perform system composition using the software reuse library, the revised requirements, and the application architecture defined in the previous section.

1. Collect Components

2. Record Issues, Trade-offs and Design Rationale

3. Link Artifacts

4. Integration Testing

5. Taking the Prototype to the Actual System

6. Component Adaptation

D. How Reuse Fits their Process

The students will discuss their current system and software development processes. The instructor will help them identify how reuse can be incorporated into their specific processes.

4.2.4 Summary

The instructor will:

summarize the important software reuse activities that have been introduced to the student.

encourage discussion and clarify any issues raised by the students.

stress the lessons learned during the students' hands-on activities.

This summary will reiterate all the important tasks and how they affect the student.

4.2.5 Lessons Learned

A. Points Confirmed

1. Improved Effectiveness and Efficiency of System Development
2. Reduced System Development Time (testing time and integration)
3. Provided Means to Rapid Prototype System
4. Helped Clarify Requirements (constraints and interfaces)
5. Built Trust in Domain Analysis Products
6. Made Job Easier

B. Factors for Failure

1. Lack of Management Support for Education and Training
2. Lack of Company Incentives
3. No Quality Assurance for Evaluating Reuse Integration into System Development
4. Tools Supporting Reuse not Provided
5. Lack of System Engineer Motivation

4.3 TRAINING INSTRUCTOR

4.3.1 Job Description

The Instructor is responsible for conducting a tailored implementation of the recommended course content through a lecture/workshop format.

This is accomplished by completing the following tasks:

consulting the reference documents listed in Section 4.4;

updating the recommended outline of course content;

preparing appropriate training materials;

conducting the training session;

developng lab exercises;

ensuring that the atmosphere of the training session is informal and relaxing, intellectually stimulating, and learner-centered;

integrating lecture, discussion, and small working groups into the training session;

leading the course completion criteria workshop; and

providing an evaluation of the course completion criteria.

4.3.2 Formal Education

The Instructor should hold at least a BS degree in system/software engineering (an MS degree is preferred) or a degree in computer science or a closely related field.

4.3.3 Knowledge of Instruction

The Instructor should possess an understanding of the principles of learning and the methods of teaching, an ability to apply these principles and methods, and excellent communication skills.

4.3.4 Practical Teaching Experience

The Instructor should have at least two years of experience in planning and conducting short courses and should possess teaching experience (at least two years), and should possess experience teaching system and software engineers.

4.3.5 Knowledge of Subject

The Instructor should possess:

experience in the system engineer processes;

knowledge and an understanding of the impact of integrating reuse into these processes;

formal training in the form of a workshop or seminar on domain analysis and domain-specific reuse;

practical experience in reuse library operations; and

work experience as system engineer or software engineer.

4.4 BIBLIOGRAPHY FOR DEVELOPING THE SYSTEM/SOFTWARE ENGINEERS COURSE

The Training Instructor should consult the following documents in preparing the tailored implementation of the course content and the training materials.

Basili, V., Caldiera, G., and Cantone, G. *A Reference Architecture for the Component Factory.* ACM Transactions on Software Engineering and Methodology, Vol 1 No. 1, Jan 1992.

Bell, T. *'90s employment: some bad news, but some good.* IEEE Spectrum, Dec 1990.

Biggerstaff, T. *Topics In Reuse and Design.* IEEE Video, 1991.

Biggerstaff, T., and Perlis, A. *Software Reusability Concepts and Models.* Vol I, ACM Press, 1989.

Cohen, S. *Modeling Software Reuse Technology: Feature Oriented Domain Analysis (FODA).* SEI, Carnegie Mellon University, May 1992.

DiTomaso, N., and Farris, G. *Diversity in the High-Tech Workplace.* IEEE Spectrum, Jun 1992.

Fairley, R., and Freeman, P. *Issues in Software Engineering Education.* Springer-Verlag, 1989.

Feiler, P. *Configuration Management Models in Commercial Environment.* SEI-91-TR-7, SEI, Carnegie-Mellon University, Mar 1991.

Frakes, W., Prieto-Diaz, R., and Arnold, R. *Software Reuse, Domain Analysis, and Reengineering.* Reston, Virginia, 6-8 April 1992.

Freeman, P. *Tutorial: Software Reusability.* IEEE Computer Society Press, 1987.

Gause, D. and Weinberg, G. *Exploring Requirements Quality Before Design.* Dorset House Publishing, 1989.

Glass, R. *Building Quality Software.* Prentice Hall, 1992.

Gomaa, H. Kerschberg, L., Bosch, C., Sugumaran, V. and Tavakoli, I. *A Prototype Software Engineering Environment For Domain Modeling and Reuse, Sixteenth Annual Software Engineering Workshop.* NASA Goddard Software Engineering Laboratory, Dec 1991.

Hooper, J., and Chester, R. *Software Reuse Guidelines.* AIRMICS, 13 Dec 1989.

Matsumoto, Y. *Some Experience in Promoting Reusable Software: Presentation in Higher Abstract Levels.* IEEE Transaction on Software Engineering, Vol SE-10 No 5, Sep 1984.

Pressman, R. *Software Engineering A Practitioner's Approach.* McGraw-Hill, 1987.

Prieto-Diaz, R. *Implementing Faceted Classification for Software Reuse.* Communications of the ACM, Vol 34 No 5, May 1991.

Prieto-Diaz, R., and Arango, G. *Domain Analysis and Software Systems Modeling.* IEEE Computer Society Press, May 1991.

Sommerville, I. *Software Engineering.* Addison-Wesley, 1989.

Tomayko, J. *Software Engineering Education.* SEI Conference 1991, Springer-Verlag, Oct 1991.

Tracz, W. *Ada Reusability Efforts: A Survey of the State of the Practice.* Proceedings of the Joint Ada Conference, Fifth National Conference on Ada Technology and Washington Ada Symposium, US Army Communications-Electronics Command, Fort Monmouth, New Jersey.

Tracz, W. *Tutorial: Software Reuse: Emerging Technology.* IEEE Computer Society Press, 1990.

Tracz, W., and Coglionese, L. *Domain-Specific Software Architecture Engineering Process Guidelines.* ADAGE-IBM-92-02, Ver 0.1, IBM Corporation, 17 Mar 1992.

Withey, J. *Model-Based Engineering.* SEI Tutorial slides, Mar 1992.

Yourdon, E. *Modern Structured Analysis.* Yourdon Press Computing Series, 1989.

Training Course
for Universities

5 TRAINING COURSE FOR UNIVERSITIES

The main objective of this course is to provide university professors with an understanding of software reuse, and to identify potential avenues for integrating domain-specific reuse into university courses and curricula.

5.1 COURSE CHARACTERISTICS

5.1.1 Rationale

Reuse is a recurring topic which must be addressed throughout the computer science curriculum. The incorporation of reuse into the current curriculum will provide students entering the work force with an awareness of the benefits and application of reuse. This course will provide support for the integration of reuse concepts into university curricula.

5.1.2 Course Objectives

The learning objectives for this course are to provide professors with an understanding of reuse, and domain-specific reuse; and to identify potential avenues for integrating domain-specific reuse into their courses and curricula.

5.1.3 Course Completion Criteria

The course completion criteria allows for small interactive working groups to apply concepts and techniques identified by the course's content areas, producing products as proof of completing the training course.

Upon completion of the training course the participants will be able to:

understand where analysis techniques, reuse concepts, and model engineering fit into the current computer science curriculum;

identify courses where reuse may be integrated;

outline specific procedures for implementing reuse across the curriculum; and

formulate a "strawman" curriculum with course descriptions.

5.1.4 Audience Characterization

This course is intended for college and university faculty who possess an interest in:

the concept of reuse;

remaining competitive with other universities;

new research topics; and

integrating reuse into their software engineering instruction.

The intended audience is responsible for curriculum development, appropriations and developing hiring guidelines. It is assumed that the participants have a software engineering background.

5.1.5 Student Readings

Arnold, R.S. *Heuristics for Aalvaging Reusable Parts from Ada Source Code.* Ada Reuse Heuristics, 90011-N, Software Productivity Consortium, Mar 1990.
Braun, C. *Ada Reusability Guidelines.* Softech, Apr 1985.

Computing Curricula 1991 - Report of the ACM/IEEE-CS Joint Curriculum Task Force. ISBN 0-8186-2220-2, Mar 1991.

Frakes, W.B. *An Empirical Framework for Software Reuse Research.* Proceedings of the Third Workshop on Methods and Tools for Reuse, No 9014, Syracuse University CASE Center, 1990.

Freeman, P. *Reusable Software Engineering: Concepts and Research Directions.* Workshop on Reusability in Programming, ITT Programming, Sep 1983.

McClure, C. *The Three R's of Software Automation: Reengineering, Repository, Reusability.* Prentice Hall, 1992.

Pressman, R.S. *Software Engineering: A Practitioner's Approach.* McGraw Hill, 1992.

5.2 OUTLINE OF COURSE CONTENT

The following is an outline of the recommended course content for Universities.

5.2.1. Reuse as Part of the Solution

A. The Dilemma

1. Software industry

   a. Increasing software costs versus hardware costs
   b. Increasing complexity of software
   c. Increasing development time
   d. Shortage of experienced personnel

2. Customer

   a. Demand for high quality products
   b. Demand for low risk products
   c. Demand for automated programming support
   d. Demand for customized software packages

B. The Solution

1. What is Reuse?

a. The process of incorporating into the life-cycle of a software system any preexisting components (e.g, requirements, design, code, executables).
b. Examples

2. Why Reuse?

a. Reduces schedule
b. Increases productivity

(1) Amplifies programming capabilities
(2) Reduces the amount of documentation and testing

c. Increases quality

(1) Software is well designed (for reuse)
(2) Software is well documented (standard)
(3) Software is well tested (certified)
(4) Software is well understood (functionality)
(5) Concepts are manifested in rapid prototyping

d. Increases reliability
e. Reduces maintenance costs
f. Overall, reuse allows for more system development within the same time frame
g. Reduces development costs

3. Evolution of Reuse

a. Early reuse projects
b. Reuse successes
c. Current approaches and concepts

(1) Components and wide spectrum reuse
(2) Composition versus generation
(3) Design for reuse
(4) Design with reuse
(5) Life-cycle importance of reuse

5.2.2 Introduction to Domain-specific Reuse

A. What is a Domain?

1. A domain is a set of common capabilities and data which constitute a set of current and future systems in a particular application area, such as:

a. A business area
b. A software business area
c. A software intensive application area
d. An application area for which similar software systems have been built

2. Examples

B. What is Domain-specific Reuse?

1. Reuse of ideas, knowledge, artifacts, personnel, and components in an existing domain.
2. Examples

C. Why Domain-specific reuse?

1. Reuse is more effective in a narrow, well defined domain where similar systems are built.
2. Examples

D. Domain Selection Checklist

1. Define strategy.
2. Select domain analysis resources.
3. Identify boundaries/scope of domain.
4. Select mature, stable, well-defined domain.
5. Define predictable technology.
6. Perform market evaluation.
7. Identify available domain expertise.
8. Conduct readiness cost and benefit analysis.

E. Existing Technological Support

1. STARS efforts
2. CARDS
3. PRISM
4. RAPID
5. Others

5.2.3 Reuse and the Software Life-cycle

A. Design for Reuse

Build into the design and implementation the flexibility, variations and adaptability to create a reusable component.

B. Design with Reuse

Incorporate available reusable assets into system analysis and design.

C. Entry-points of Life-cycle Artifacts

It is important to stress that reusable components include all life-cycle artifacts such as requirements, design, code, test suites.

D. Reengineering

1. When to reengineer
2. Economic considerations
3. Data reengineering and software reengineering
4. Benefits of reengineering

a. Reduced maintenance
b. Upgrading software along with upgrading software engineering practice
c. Easier documentation and testing

5. Incremental reengineering

E. CASE

1. Features of CASE tools
2. CASE Issues
3. Sample CASE tools
4. CASE tools as a research topic

5.2.4 Considerations When Integrating Reuse

A. Economic issues

A university must consider the costs associated with incorporating reuse.

1. How much will it cost to implement reuse into the curriculum?
2. Component availability
3. Library Maintenance and availability
4. Is new hardware required?
5. Network connection cost
6. What are the alternatives?

B. Feasibility analysis

A university must decide if it is feasible to incorporate reuse into their current curriculum.

1. Is the reuse program supported by administration?
2. Does the school have the resources to implement the program?
3. Does the department have personnel with the appropriate training?
4. What is the scope of reuse in the curriculum?

C. Administrative/Departmental Issues

1. Departmental support

a. Professors
b. Instructors
c. Assistant chairman
d. Chairman
e. Does the department have a long term committment to reuse?

2. Administrative support

a. Dean of college
b. Dean of academic affairs
c. Is the administration aware of the benefits of reuse?

3. Technical aspects

   a. Can existing hardware be used?
   b. What networks need to be accessed?
   c. What library mechanisms are required?
   d. What other software is required (CASE tools)?
   e. Alternatives

4. Incremental in-house reuse

   a. Software engineering
   b. Design for reuse and data abstraction
   c. System analysis
   d. Domain analysis
   e. Put components designed for reuse into in-house reuse library
   f. Design with reuse using existing in-house reuse library
   g. Introductory courses and reuse concepts

5.2.5 Continuing Education/Research

A. Does the staff have interest in reuse as a reseach topic?
B. Is there funding available from government/ industry to support research?

5.2.6 Summary of Content

A. Firm commitment from staff and administration
B. Assess benefits of introducing reuse into the curriculum
C. Identify and resolve impediments to the introduction of reuse into the curriculum

5.3 TRAINING INSTRUCTOR

5.3.1 Job Description

The Training Instructor is responsible for conducting a tailored implementation of the recommended course content through a lecture/workshop format.

This is accomplished by completing the following tasks:

consulting the documents listed in Section 5.4;

updating the recommended outline of course content;

preparing appropriate training materials;

estimating the duration of the training session based on a blend of optimum cost-effectiveness and covering the course content thoroughly;

conducting the training session;

seeing that the atmosphere of the training session is informal and relaxing; intellectually stimulating; and learner-centered;

integrating lecture, discussion, and small working groups into the training session;

leading the course completion criteria workshop; and

providing an evaluation of the course completion criteria.

5.3.2 Formal Education

The Instructor should hold a PhD in computer science, software engineering, system engineering, or a closely related field.

5.3.3 Knowledge of Instruction

The Instructor should possess an understanding of the principles of learning; the methods of teaching; an ability to apply these principles and methods; and excellent communication skills.

5.3.4 Practical Teaching Experience

The Instructor should possess teaching experience (at least two years), experience in planning short courses, and in addressing university faculty.

5.3.5 Knowledge of Subject

The Instructor should possess formal training in the form of a workshop or seminar on domain analysis and domain-specific reuse, and work experience in software development, domain analysis, and reengineering.

5.4 BIBLIOGRAPHY FOR DEVELOPING UNIVERSITY COURSE CONTENT

The Training Instructor should consult the following documents in preparing the tailored implementation of the course content and the training materials.

Matsumoto, Y. *Some Experience in Promoting Reusable Software: Presentation in Higher Abstract Levels.* IEEE Transaction on Software Engineering, Vol SE-10 No 5, Sep 1984.

Pressman, R. *Software Engineering A Practitioner's Approach.* McGraw-Hill, 1987.

Tomayko, J. *Software Engineering Education.* SEI Conference 1991, Springer-Verlag, Oct 1991.

Tracz, W. *Ada Reusability Efforts: A Survey of the State of the Practice.* Proceedings of the Joint Ada Conference, Fifth National Conference on Ada

Technology and Washington Ada Symposium, US Army Communications-Electronics Command, Fort Monmouth, New Jersey.

6 EXECUTING THE TRAINING PLAN

There are two primary methods by which the Training Plan can be effectively executed.

6.1 ONE COURSE AT A TIME

The first method is to conduct each course independent of the other courses, i.e. one course at a time (see Figure 1.1). This is accomplished by appointing a Training Supervisor to oversee the training course's management and a Training Instructor to conduct the actual training session.

All of the necessary information for pursuing this strategy is encapsulated for each training course in its respective section in this document. This allows for each training course to be independently removed from the Training Plan and forwarded to perspective implementors without any required knowledge of the other training courses.

6.2 INTEGRATED PACKAGE

The second method is to conduct each course as an integral part of a total package. This requires appointing a Training Executor to oversee the top-level management of this strategy (see Figure 1.2).

The Training Executor identifies organizations that have a working relationship. The Training Supervisors and Training Instructors function in the same manner as they would in conducting independent versions of their courses. The Executor then formulates a coherent set of instructional courses for each audience.

Figure 1.2 illustrates how the System and Software Engineers training course becomes the main thread of the Training Plan through this integrated strategy. Of the two methods for executing the Training Plan, the integrated package will have a stronger impact on integrating reuse and a higher potential of accomplishing a shift within DoD and industry to reuse-based domain-specific software development.

The necessary information for pursuing this strategy is a combination of each of the four training courses and the administrative functionality of the Training Executor. Each training course becomes a tailored implementation of the Training Plan based not only on the target audience but also on considerations of the other audiences.

Figure 1.1: Execution of Training Plan

Figure 1.2: Integrated Package

6.3 TRAINING EXECUTOR

The Training Executor's functionality is critical for the successful implementation of the Training Plan as an integrated package. To ensure proper and efficient implementation of the Training Plan, the Training Executor is responsible for completing the following tasks:

1. Identify DoD organizations, DoD contractors, and universities that have previously established a working relationship with each other. These audiences could also be identified from a geographical perspective to assist in building such working relationships.

2. Select an appropriate training facility available to all audiences, if applicable.

3. Coordinate the efforts of Training Supervisors.

4. Assure the correct implementation of the Training Plan based on the established working relationship between DoD organizations, DoD contractors, and universities.

5. Monitor the impact of the training across the three audiences.

6.4 TRAINING SUPERVISOR

Functionality is critical for the successful execution of the training course. The time period for completing the following tasks is approximately 6-8 weeks.

The Training Supervisor is responsible for:

describing the intent of the course to the Training Instructor;

determining the optimum number of participants;

scheduling the training sessions by coordinating the facility's availability with the Training Instructor;

addressing food, lodging, and transportation;

registering the participants;

forwarding materials to each participant;

coordinating all efforts between the Training Instructor, facility staff, and Training Executor;

assuring the correct implementation of the recommended course content by the Training Instructor;

securing training equipment;

reproducing training materials;

administering evaluations of the training (see APPENDIX B);

producing a lessons learned document containing information that may be used in updating the TP and/or the course content at a later date; and

documenting the impact of the training by tracking requests for further training and requests for further information.

6.5 TRAINING FACILITY

The style of the classroom must be established. Some factors to consider:

lecture versus conference style room;

seating capacity;

computer resources; and

proximity to convenient facilities.

Training devices appropriate to the types of training materials must be available.

APPENDIX A PRICE WORKSHEET

The following is a Price Worksheet to assist in predicting dollar amounts for conducting the training courses.

A. Price Worksheet

1. Administration
2. Instructional Time/Travel Costs
   a. Supervisor
   b. Instructor
   c. Technician
   d. Participant Time/Travel Costs
3. Scheduling Training Session
4. Registration of Participants
5. Site Set-up
6. Refreshments
7. Communications

B. Course Materials

1. Preparation for Training Instructor
   a. Agenda
   b. Course outline
   c. Handouts
   d. Transparencies/Overheads
2. Reproduction for Participants
   a. Paper copies of slides
   b. Lab worksheets
   c. Handouts
   d. Course evaluation forms
   e. Proceedings

C. Support Costs

1. Facilities
2. Equipment

APPENDIX B COURSE EVALUATION FORM

The following are the proposed contents of Course Evaluation Form to be administered to the participants after the training session is completed. It is recommended that the Training Supervisor administer the form, or appoint someone in his absence.

PART I. STUDENT INFORMATION

1. Name
2. Job Title
3. Date
4. Course Dates
5. Number and Title of Course
6. Training Location

PART II. COURSE MATERIAL

1. Applicability of Information Received in Course
2. Technical Value of Information Received in Course
3. Length of Course
4. Quality of Training Materials Used in Course

PART III. COURSE PRESENTATION

1. Quality of Training
2. Quality of Presentations
3. Training Instructor's Knowledge of the Subject
4. Training Instructor's Ability to Stick to the Subject
5. Amount of Theory
6. Amount of Practice or Application
7. Time Devoted to Content

PART IV. TRAINING AIDS

1. Use
2. Quality
3. Quantity

PART V. TOOLS AND EQUIPMENT

1. Quantity

2. Quality

PART VI. OVERALL OPINION OF TRAINING COURSE

PART VII. PROBLEMS ENCOUNTERED

PART VIII. RECOMMENDATIONS FOR TRAINING COURSE

PART IX. SIGNATURE (optional)

APPENDIX C GLOSSARY

Analysis: the separation of any material or abstract entity into its constituent elements. This process has a method of studying the nature of something or of determining its essential features and their relations.

Application Domain: the knowledge and concepts that pertain to a particular computer application. Examples include battle management; avionics; communication, command, control and intelligence; and nuclear physics.

Architecture: a template that models the system which includes the user interface, input, system function and control, output, and maintenance and self test as elements. The architectural template allows the analyst to create a hierarchy of detail.

ASR: Ada Software Repository.

Asset: a reusable entity.

ASSET: Asset Source for Software Engineering Technology.

CARDS: Central Archive for Reusable Defense Software.

Classification: the process of organizing assets according to a defined classification method.

Classification Methods: defined methods of classifying assets supported by a library mechanisms.

Classification Scheme: an organizational approach to utilization of classification methods.

Context Diagram: establishes the information boundary between the system being implemented and the environment in which the system is to operate.

Control Flow: shows the control information passed throughout the system and associated control processing.

Cost: includes manpower, skills and availability, risk to the schedule budgets and new software and hardware development budgets.

Data Flow: shows the data information passed throughout the system as it moves from input to output.

Domain: the set of current and future applications marked by a set of common capabilities and data. An area of activity or knowledge. Domains can be characterized as application, solution, horizontal, or vertical.

Domain Analysis: the process of identifying objects and operations of a class of similar systems in a particular problem domain; pertains to operational software reuse libraries.

Domain-specific Reuse: reuse of ideas, knowledge, artifacts, personnel, and components in an existing domain.

DSRS: Defense Software Reuse System

Entity Relationship Diagram: a modeling tool used to portray the relationships that exist among data stores.

Hierarchical Model: a top down view of a system.

Horizontal Domain: the knowledge and concepts that pertain to a particular functionality of a set of software components that can be utilized across more than one application domain. Examples include user interfaces, database systems, and statistics. Most horizontal domains can be organized as a set of equivalence classes where the distinguishing characteristics are software decomposition style (functional, object-oriented, data-oriented, control-oriented, declarative, etc.), conceptual underpinning, and/or required hardware. One example is subdividing user interfaces into ANSI terminal supporting versus bit-mapped, mouse input supporting.

Library Mechanism: an automated tool that supports classification, search and retrieval of assets.

Model Engineering: developing abstract representations of what will eventually become a combination of computer hardware and computer software.

RAPID: Reusable Ada Packages for Information System Development

Rapid Prototyping: a process that enables the developer to create a model of the system to be built. The model itself may not evolve into the system.

Software Engineer: software engineers are concerned with how the individual system components are actually designed, developed, and integrated into the total system.

Software Reuse: the process of incorporating into a software system any preexisting component generated at any stage of a system's development.

Solution Domain: the knowledge and concepts that pertain to a particular software solution for providing functionality desired in an application system. Examples include graphical user interfaces, relational database systems, client-server architectures, and expert system shells.

State Transition Diagrams: a diagram that highlights the time-dependent behavior of the system.

System Engineer: called a system analyst in some application domains; begins with customer-defined goals and constraints and derives a representation of function, performance, hardware and software interfaces, design constraints, and information structure that can be allocated to each of the generic system elements.

Vertical Domain: The knowledge and concepts that pertain to a particular application domain. Vertical domains can be organized as a hierarchy of subdomains that specialize the knowledge and concepts as one moves from the root to the leaves. Most application domains can be organized into a vertical domain hierarchy.